GANCCRobot

Wu, Ruiqi; Zhou, Changle ; Chao, Fei; Yang, Longzhi ; Lin, Chih-Min; Shang, Changjing

Published in:
Information Sciences

DOI:
10.1016/j.ins.2019.12.079

Publication date:
2020

Citation for published version (APA):
Wu, R., Zhou, C., Chao, F., Yang, L., Lin, C-M., & Shang, C. (2020). GANCCRobot: Generative Adversarial Nets based Chinese Calligraphy Robot. *Information Sciences*, *516*, 474-490. https://doi.org/10.1016/j.ins.2019.12.079

Highlights

GANCCRobot: Generative Adversarial Nets based Chinese Calligraphy Robot

Ruiqi Wu, Changle Zhou, Fei Chao, Longzhi Yang, Chih-Min Lin, Changjing Shang

- This project developed Generative Adversarial Nets for the implementation of a calligraphy robot.
- The robot can control the type and style of strokes through the proposed model.
- The robot can write strokes with good quality that is close to the human level.
- Latent variables in the proposed model improve the diversity of styles of strokes.
- Our model is superior to the other models regarding the quality and diversity of strokes.

GANCCRobot: Generative Adversarial Nets based Chinese Calligraphy Robot

Ruiqi Wu[a], Changle Zhou[a], Fei Chao[a,b,*], Longzhi Yang[c], Chih-Min Lin[d], Changjing Shang[b]

[a]*Department of Artificial Intelligence, School of Informatics, Xiamen University, China*
[b]*Institute of Mathematics, Physics and Computer Science, Aberystwyth University, UK*
[c]*Department of Computer and Information Sciences, Northumbria University, UK*
[d]*Department of Electrical Engineering, Yuan Ze University, Taiwan*

Abstract

Robotic calligraphy, as a typical application of robot movement planning, is of great significance for the inheritance and education of calligraphy culture. The existing implementations of such robots often suffer from its limited ability for font generation and evaluation, leading to poor writing style diversity and writing quality. This paper proposes a calligraphic robotic framework based on the generative adversarial nets (GAN) to address such limitation. The robot implemented using such framework is able to learn to write fundamental Chinese character strokes with rich diversities and good quality that is close to the human level, without the requirement of specifically designed evaluation functions thanks to the employment of the revised GAN. In particular, the type information of the stroke is introduced as condition information, and the latent codes are applied to maximize the style quality of the generated strokes. Experimental results demonstrate that the proposed model enables a calligraphic robot to successfully write fundamental Chinese strokes based on a given type and style, with overall good quality. Although the proposed model was evaluated in this report using calligraphy writing, the underpinning research is readily applicable to many other applications, such as robotic graffiti and character style conversion.

*Corresponding author

Email addresses: wuruiqi@stu.xmu.edu.cn (Ruiqi Wu), dozero@xmu.edu.cn (Changle Zhou), fchao@xmu.edu.cn (Fei Chao), longzhi.yang@northumbria.ac.uk (Longzhi Yang), cml@saturn.yzu.edu.tw (Chih-Min Lin), cns@aber.ac.uk (Changjing Shang)

Keywords: Calligraphy robot, generative adversarial nets, motion planning

1. Introduction

Robotics has been widely applied to promote human culture and education, such as robotic Chinese character writing [1, 2], dancing, and drawing. Robotic writing is a particularly hot topic due to the great applicability of its key technology in other applications, including robotic drawing [3], industrial welding [4, 5], and medical rehabilitation [6] among others. The essence of robotic writing is the generation of sequences of robotic actions in accordance with human evaluation criteria. Thus, the focus of recent research is the design of control algorithms to drive robotic end-effectors to write complex characters or letters [7, 8]. It is generally more challenging for a robot to write Chinese characters than western letters, because a Chinese character often consists of many strokes, each of which must be placed in a specific position with specific size [9, 10]. Therefore, the writing quality of Chinese characters fundamentally depends on the quality of its comprising strokes. Two challenges still remain to be resolved in order to generate high-quality strokes and thus characters: 1) restricted stroke diversity limited by the training samples [11, 12], and 2) difficulty in designing the evaluation mechanism in generating strokes [13].

The first challenge is often handled by the employment of large stroke datasets to train robot control systems [14]. These methods indeed improves the style diversity of the generated strokes to some extent, but the challenge remains as the styles are still tied with the training samples [15, 16]. Notice that calligraphers break through the limitation of learning samples and create new writing styles through human creativity, and human-computer interaction methods have been widely used in robotic control [17, 18]. Therefore, the follow-up ability of manipulators has been applied to generate new writing styles that are learned by robots directly from human demonstrators [19, 20, 21]. Despite the success in learning new writing styles that do not exist in the sample library, such methods usually entail significant human work to enable manipulators to produce sufficient style information, which is a typical drawback of open-loop robotic systems [22, 23].

Different evaluation mechanisms have been adopted to robotic writing systems based on either an open- or closed-loop structure [24]. In particular, human expertise is often utilized in open-loop systems to evaluate the generated results, which is very labor intensive [25, 26]. An evaluation function

is typically employed in a closed-loop robot system, which is designed by algorithm engineers based on stroke features and human aesthetic mechanisms [27]. Such process, again, requires significant human effort, in addition to aesthetic knowledge of calligraphy. Furthermore, due to the subjectivity of calligraphy art, many evaluation criteria are difficult to be regularized. As a result, it is extremely difficult for humans to design an evaluation function that can accurately measure the quality level of calligraphy. This leads to the demand of an automatic learning mechanism to build the evaluation function.

This paper proposes a novel robotic writing approach based on the Generative Adversarial Nets (GAN) [28] in an effort to address the aforementioned challenges, which allows the robot to learn to write and control the styles and types of fundamental Chinese strokes. The GAN model, as a semi-supervised learning approach, is structurally simple and functionally powerful in terms of diversity, which has been widely applied in computer vision. Thanks to such great features, the GAN model is employed in this work to support robotic calligrapher to learn new writing styles. To control the type of the generated strokes, the label information of the stroke is provided to the GAN model so that the robot can write the stroke according to the stroke label. In addition, inspired by the InfoGAN [29], a set of latent codes is added at the input of the proposed model, which learns the style features of strokes by maximizing mutual information between the latent codes and the generated strokes.

Different from the traditional robotic calligraphy models, the proposed robot system is implemented based on a modified GAN model. Thanks to the introduction of the modified GAN model, the proposed robot system can control the type and style of the generated strokes, as demonstrated and verified by a series of experiments in Section 4. The main contribution of this work includes: 1) implementing a new GAN model that can handle the dominant features and recessive features of data, with theoretical and empirical proof; and 2) developing a Chinese calligraphy robot based on the proposed GAN model that is able to produce various types of writing strokes without the common requirement of an explicit evaluation function.

The remainder of this paper is organized as follows: Section 2 introduces the related background of the proposed model, including the GAN model with the support of mutual information theory, and the calligraphy robot system. Section 3 describes the proposed model which can write strokes according to the specified stroke labels and styles. Section 4 details the

experimentation and discusses the experimental results. Finally, Section 5 provides a brief conclusion and directions for future work.

2. Background

2.1. Generative Adversarial Nets and Variants

The GAN model owns good performance in data distribution learning tasks [28]. A typical GAN model consists of two competing networks: the generative network and discriminative network. The function of the discriminative network is to determine whether the sample x is within the real data distribution P_r. Therefore, the goal of the discriminative network can be expressed as:

$$\max_{\phi} \; E_{x\sim P_\theta}[\log(1 - D_\phi(x))] + E_{x\sim P_r}[\log(D_\phi(x))], \tag{1}$$

where D_ϕ is an appropriate real-valued function parameterized by ϕ, and P_θ denotes a data distribution that is mutually exclusive with P_r (i.e., $P_\theta \bigcap P_r = \emptyset$). On the contrary, the function of the generative network is to convert the input z that obeys the distribution P_z into fake data $\hat{x}$ that obeys the distribution P_f. The goal of the generative network is to drive the distribution P_f infinitely approaching to the distribution P_r. The Jensen-Shannon divergence can be used to measure the distance between two distributions [30]; thus, the goal of generative network can be essentially transformed to minimizing the Jensen-Shannon divergence between P_f and P_r. If $P_f = P_r$, the Jensen-Shannon divergence is minimized, indicating the generative network perfectly replicate the real data distribution P_r. The goal of generative model is expressed as:

$$\min_{\phi} \; \mathbb{E}_{z\sim P_z}[\log(1 - D(G_\phi(z))], \tag{2}$$

where $G_\phi(z)$ denotes the fake data generated by the generation model according to the noise z.

The generative and discriminative networks are usually optimized by the Minimax algorithm. The working process of the GAN can be expressed as that: the generator takes noise as input and generates samples, and the discriminator receives samples from both the generator and training dataset and then distinguishes between the two sources. These two networks play a continuous game, where the generator learns to produce more and more

realistic samples, and the discriminator learns to become more and more powerful in distinguishing the generated data from the real ones. These two networks are trained simultaneously, with the expectation to generate indistinguishable samples from real data.

A large number of variations have been proposed in the literature due to its great success, such as Wasserstein GAN, Condition GAN [31], SeqGAN [32] and InfoGAN [29]. Different from the original GAN, the label of each sample in the Condition GAN is added to the inputs of the generator and discriminator. Thus, the Condition GAN can control the type of the generator output. Unlike the explicit control of the Condition GAN, the infoGAN implicitly controls the results of the generator by controlling a part of the input noise. In particular, the InfoGAN divides the input noise from the generator into two parts, and maximizes the mutual information between one part of the noise and the output of the generator through an auxiliary distribution network. When the part of the noise and the output of the generator have high mutual information, the part of the noise can be regarded as partial feature representation of the output of the generator. Thus, when the value of the part of the noise is changed, some features of the output of the generator will be changed accordingly. In addition, GAN has also been combined with other algorithms to expand its application range. For example, as a combination of GAN and expectation-maximization (GAN-EM) learning approach, GAN-EM can be used for clustering, semi-supervised classification and dimensionality reduction [30]; in particular, GAN-EM can achieve state-of-the-art clustering and semi-supervised classification results.

The GAN and its variants have been widely applied, including in the filed of computer painting and robotic calligraphy [33]. In particular, GAN and auto-encoder has been combined for the synthesis of Chinese calligraphy [34]. This only solved the image-to-image conversion problem and did not work for image-to-action conversion. The task of the robot writing Chinese characters can be regarded as a robot that generates and performs writing actions based on the provided image, so as to write on the canvas. Because the calligraphy is artistic, it is difficult to establish a suitable aesthetic evaluation standard. Therefore, it is a challenging task to create a proper evaluation mechanism for robotic calligraphy writing behaviors. However, the discriminate network in GAN is able to learn aesthetic criteria entailed in the training data through the adversarial training phase, and actually, it has been attempted to apply GAN into the robot calligraphy in the work of [35]. In this work, although the robot can learn to write strokes, but the type and style of the output

strokes cannot be controlled.

2.2. Robotic System

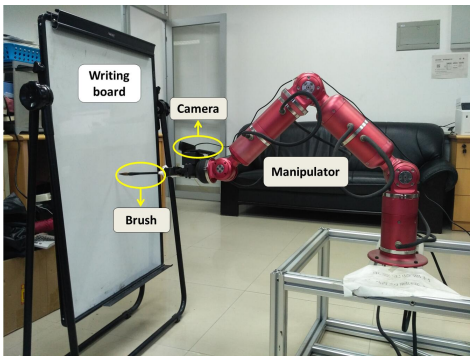
The robotic system used in this work, similar to most of the calligraphy robots, is comprised of a robotic arm with a pen [12, 36, 37]. The robot's arm uses the pen to write characters on a white board or a piece of paper. Fig. 1a shows the hardware of the calligraphy robot system, which includes a 5-DOFs industrial robot arm, a calligraphy brush, a camera, and a writing board. The robot arm is fixed in a position, and the white board is placed in front of the robot arm. A brush is attached to the robot's hand, and a camera is mounted above the brush. When the robot writes a character on the white board, the camera captures a picture of this character and converts it into a binary image.

Four joints of the arm are used to achieve the writing task. The writing activity happens within the working range of the arm. The kinematic configuration of these four joints is shown in Fig. 1b. The four joints are represented by j_1, j_2, j_3, and j_4. The links between joints are represented as L_I, l_2, l_3, and l_4, with lengths of $150mm$, $375mm$, $354mm$, and $175mm$, respectively. Having known the hardware information and the forward kinematics, the inverse kinematics function can be calculated. After calculating the trajectory of the brush movement for a given character, the angular values of the four joints of the robot can be determined by inverse kinematics.

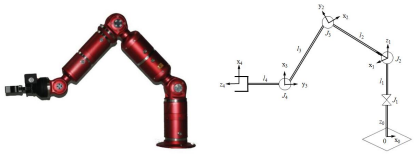
3. Proposed Approach

3.1. The Approach Overview

It has become a common practice to apply neural networks to robotic control to meet various real-world demands [38, 39]; and this work also follows this general practice, by focusing on robotic calligraphy using the GAN. The framework of the proposed method is illustrated in Fig. 2, which is comprised of four parts: (1) a stroke generative module, (2) a stroke discriminative module, (3) an auxiliary distribution module, and (4) a calligraphy robot system. The task of the generative module is to generate stroke trajectory points based on certain given input information that includes a random noise, z, stroke type information, l, and latent codes, c. The task of the discriminative module is to correctly classify the samples, distinguishing them between those from the generative module and the rest from the real training



(a) The robotic system for writing Chinese strokes.



(b) The kinematic configurations of the experimental manipulator.

Figure 1: The robotic system.

data. The auxiliary distribution module is used to maximize the mutual information between the input latent codes c, and the stroke trajectory points generated by the generative module. The calligraphy robot system has two functions: (1) writing a stroke using the stroke trajectory points; (2) converting the stroke to an image.

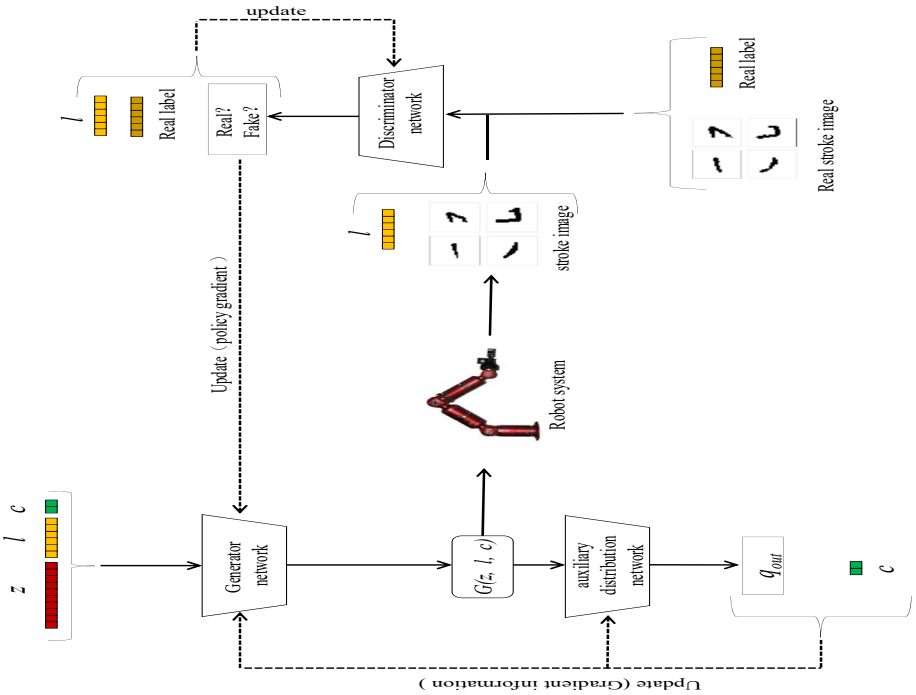


Figure 2: The flowchart of the proposed approach for robotic handwriting. The solid line shows the flow of data and the dotted line shows the direction of the gradient information.

The training objectives of the model are summarized as follows: (1) to train the discriminative module to minimize the classification error rate, (2) to train the auxiliary distribution module to maximize the mutual information between the specified latent codes c, and the output of the generative module; (3) to train the generative module to generate the stroke trajectory points such that the trajectory is difficult for the discriminative module to classify and has the implicit features expressed in the latent codes c.

Different with the traditional GAN model, the output of the generative module in the proposed approach cannot be directly used as the input of the discriminative module, but the images of the generated strokes are the

input of the discriminative module. The robot is required to learn the calligraphy writing behaviors from the calligraphy images in this work; thus, the generator network needs to produce the writing action sequences and the discriminator network still takes the images as input. Consequently, the writing action sequences cannot be processed by the discriminator network. The writing action sequences generated by the generator are performed by the robot, then, the writing results are captured as stroke images by a mounted camera, which are used as fake samples for the input of the discriminator.

The proposed approach also differs from the traditional GAN model in its training stage, because the proposed model requires the robot to write strokes. The traditional GAN uses the back-propagation algorithm to update model parameters; and the generator updates its parameters by backpropagating the discriminator's predicted error. However, in the proposed model, the output of the generator cannot be directly used as the input of the discriminator, as the model itself is non-differentiable. To address this issue, the policy-gradient method employed in the work of [32] is introduced in this work to train the GAN model. Briefly, the policy-gradient method is a basic training mechanism in the reinforcement learning algorithm that can implement the error backpropagation of the discriminator on the generator with the participation of the robot.

3.2. Stroke Discriminative Module

The discriminative module is fundamentally a binary classifier, which computes the probabilities that a sample is from the training set and that the sample is generated by the generative module. This module is practically implemented as a Multi-layer Perceptron (MLP) network, D. Denote the training data set and the set of the generated images as X_{real} and X_{fake}, respectively; and denote the overall data set as X, that is $X = X_{real} \cup X_{fake}$. The traditional GAN model only takes one input which is a sample x, ($x \in X$), and produces the output which is the probability that sample x belonging to X_{real}. Differently, the input of the proposed GAN model consists of two parts; the label information about the stroke type is also included as the input of the proposed GAN model, in addition to a stroke image.

The network D consists of three layers of neurons, the input layer, the hidden layer, and the output layer. In particular, the input layer contains 794 neurons; the size of each image is 28×28, and the size of each label is 10. There are 128 neurons in the hidden layer, and there is only one neuron in the output layer. As shown in Fig. 2, each data instance in the X_{real} set

consists of a stroke image and a label of the stroke. After the robot writes the stroke according to the stroke trajectory points generated by the generative module, the output stroke image is captured by a camera mounted on the gripper of the robot. The image is then added in the X_{fake} set, while the label l is provided by system users as system input.

3.3. Stroke Generative Module

The stroke generative module generates a sequence of stroke trajectory points according to the input information. The stroke generative module is also established by an MLP network, G. The input information consists of three parts: the noise, z; the stroke label, l; the latent code, c. The added stroke type label is used to control the type of the output stroke. The added latent code is employed to control the style transformation of the output stroke, which is detailed in the Auxiliary Distribution Module. The input noise, z, is generated by a Gaussian noise generator, which is set as a 128-dimensional vector. The input stroke label is represented by one-hot encoding, with the dimension of 6. The input latent code is set as 2-dimensional continuous noise. The latent code can be regarded as controller noise.

Denote the output layer as G_{output}, each stroke trajectory point in this later contains two parts: (1) coordinate information of the point in the 28×28 coordinate system, (2) width information about the trajectory. Thus, the output of G_{output} is defined as follows:

$$G_{output} = (P_N + W_N) \cdot T_N, \tag{3}$$

where P_N denotes the coordinates of the points; W_N denotes the width of each level of the strokes in the trajectory of the manipulator of the robot; T_N denotes the number of trajectory points. In particular, W_N is set as 20 in this work. The input image of the network D is of a resolution of 784 pixels. Thus, to fully map the image coordinates, the coordinates of a trajectory point are represented as a 784-dimensional vector through one-hot encoding. Likewise, the type of trajectory width (W_N) is also represented by one-hot encoding. Because P_N and W_N belong to two different distributions, the activation values for P_N and W_N must be separately calculated using a "softmax" activation function.

The complexity of the generative module increases when the number of track points increases. Thus, the number of neurons is proportional to the

number of trajectory points in the hidden layer with the ratio of (128:1) in this work. In the proposed model, T_N is set as 5. Consequently, the number of neurons is 640. In summary, the number of neurons in the G network is set as follows: 136 neurons in the input layer, 640 neurons in the hidden layer, and 4,020 neurons in the output layer.

After the G network generates all the probability distributions of the trajectory points, a random sampling method is used to sample the trajectory points from the probability distribution of each trajectory. Then, the calligraphy robot writes a stroke based on these trajectory points.

3.4. Auxiliary Distribution Module

A latent code is introduced in this work to the input layer of the proposed generative network, inspired by the InfoGAN [29]. The latent code, as part of the input noise, is assumed here to represent some features of a stroke. Moreover, the G network requires assurance that the feature information represented by the latent code cannot be lost in generating the trajectory points. This was assured by the proposed network using an information-theoretic assumption: if the feature information is not lost, there should be high mutual information between the latent codes c and the generator distribution $G(z, l, c)$.

In information theory, mutual information is used to measure how much knowledge about a random variable, X, can be learned from another random variable, Y. If X and Y are independent, their mutual information $I(X; Y)$ is zero. In contrast, after observing variable, Y, the greater the uncertainty reduction of the variable, X, the greater their mutual information. Then, the mutual information is expressed as the difference of two entropy terms as follows:

$$I(X; Y) = H(X) - H(X|Y) = H(Y) - H(Y|X). \tag{4}$$

Denote the mutual information between the latent codes c and generator distribution $G(z, l, c)$ as $I(c; G(z, l, c))$. The proposed work also sets a variational lower bound, $L_I(G, Q)$, to approximate $I(c; G(z, l, c))$, where G is the generative network, and Q denotes an auxiliary distribution to approximate the posterior, $P(c|G(z, l, c))$. Practically, the auxiliary distribution, Q, is set as a neural network, whose input is $G(z, l, c)$ and whose output is the parameters for the conditional distribution $Q(c|G(z, l, c))$. Therefore, the auxiliary distribution Q can be interpreted as: Q restores the latent codes c, according

to the input $G(z, l, c)$. In addition, there is another function of Q, which provides the gradient information for the G network working together with the D network.

The structure of the latent codes c, consists of the latent variables $c_1, c_2, \ldots, c_L$, where L is the number of the latent codes. The values of the latent codes, c, can be continuous or discrete. In this particular work, the length of c is set as 2, and the values are continuous. This setting can better observe the impact of the latent codes on the stroke style. Hence, the auxiliary distribution network Q has an input layer with 4,020 neurons, a hidden layer with 740 neurons, and an output layer with 2 neurons.

3.5. Training Module

The training module, as shown in the dotted lines of Fig. 2, is used to train the stroke discriminative module, the stroke generative module, and the auxiliary distribution module. The policy gradient and label information are introduced into the training module, the training objective of the original GAN is defined as:

$$\begin{aligned} \min_G \max_D V(D,G) =& \mathbb{E}_{x\sim p_r}[\log D(x|l)]+ \\ & \mathbb{E}_{z\sim p_z}[\log(1-D(W(G(z|l,c))))], \end{aligned} \tag{5}$$

where x denotes the input of the D network; $G(\cdot)$ denotes the G network's output; $W(\cdot)$ denotes the writing process of the robotic system; and $D(\cdot)$ denotes the D network's output.

Due to the additions of the variational regularization and auxiliary distribution networks, the proposed model can be regarded as a minimax game of these three networks (G, Q, D). Thus, the training objective of the model is mathematically represented as:

$$\min_{G,Q} \max_D V_B(D,G) = V(D,G) - \lambda L_I(G,Q), \tag{6}$$

where λ denotes a hyper-parameter for controlling the impact of the mutual information in the game; and $L_I(G, Q)$ denotes the lower bound of the mutual information of $G(z, l, c)$ [29].

The task of the discriminative network is to identify the source of the input stroke image, x, and the label, l. Thus, the loss function of the D network is expressed as follows:

$$\begin{aligned} D_{loss} =& -\mathbb{E}_{x\sim p_r}[\log D(x|l)]- \\ & \mathbb{E}_{z\sim p_z}[\log(1-D(W(G(z|l,c))))]. \end{aligned} \tag{7}$$

The task of the Q network is to restore c; thus, the selection of the loss function in the Q network is designed in accordance with the type of values for the latent codes, c. As there are only two consecutive latent codes in the proposed model, the least squares method is employed to implement Q in this work, which is defined as follows:

$$Q_{loss} = \sum_{i=1}^{n} (c_i - Q(G(c|z,l)_i))^2, \tag{8}$$

where n denotes the size of the mini-batch training samples, and i denotes the index of the sample.

As shown in Eqs. 5 and 6, the gradient information of the G network is provided by the D and Q networks. The gradient information provided by D is a policy gradient. Then, as shown in Eq. 9, the loss function of the G network contains two parts: policy gradient information, P_{loss}, and gradient of mutual information, λQ_{loss}:

$$G_{loss} = P_{loss} + \lambda Q_{loss}, \tag{9}$$

$$P_{loss} = \mathbb{E}_{z\sim p_z, l\sim p_l, c\sim p_c}[(\log \prod_{i=1}^{n} G(z,l,c)_i) \cdot D(W(G(z,l,c)))], \tag{10}$$

where n denotes the number of trajectory points, and i denotes the index of the trajectory points.

An advanced gradient optimization algorithm, the Adam algorithm, is adopted in this work to train these three networks. The training process of the model is indicated by the dotted line in Fig. 2. The training processes in the three networks alternate according to their loss functions. The hyper-parameter, λ, is set as 1, indicating that, in each round of training, the Q network updates once the parameters of the G network are updated. Therefore, the parameters of the G network are updated twice in a round of training. In addition, another update is conducted according to the policy gradient provided by the D network. The pseudo-code of the entire training procedures is summarized in Algorithm 1.

3.6. Theoretical Analysis of Stability

Given that the generator and discriminator play a "minimax game" in their training process, the training procedure of the proposed model includes

Algorithm 1 Training procedure of GANCCRobot

Require: Real stroke image dataset (X_{real}, L_x) and random number Z, L, C;

Initialize G, D and Q with random weights;
repeat
  for g-step **do**
    Produce a set of random number z, l, c;
    Input z, l, c into G;
    Output a set of probability distribution G_{out} for trajectory points;
    for t in $1 : T_N$ **do**
      Sample a trajectory point position form trajectory point distribution;
      Sample a trajectory point width corresponding the above trajectory point;
    end for
    Robot writes the trajectory, then the writing result is converted to an image X_{fake};
    calculate P_{loss} by Eq. 10;
    Use P_{loss} as Q_{loss} to update G parameters;
  end for
  for q-step **do**
    Input G_{out} into Q;
    output q_{out}
    calculate Q_{loss} by Eq. 8;
    Update G parameters by Adam algorithm;
    Use λQ_{loss} as Q_{loss} to update G parameters;
  end for
  for d-step **do**
    Produce current G and robot to generate new trajectory images(X_{fake}, c) and combine with (X_{real}, l_x);
    Train D by Eq. 7;
  end for
until GAN Converges

two steps: 1) keep G fixed, to maximize the quantity of $V(G, D)$ to optimize D; and 2) keep D fixed, to minimize the quantity of $V(G, D)$ to optimize G.

Thus, $V(G, D)$ can be represented as:

$$\begin{aligned} V(D,G) &= \int_x p_r(x|l)\log(D(x|l))\,dx + \int_z p_z(z|l,c)\log(1-D(W(G(z|l,c))))\,dz \\ &= \int_x p_r(x|l)\log(D(x|l)) + p_g(x|l)\log(1-D(x|l))\,dx. \end{aligned} \tag{11}$$

The optimal discriminator D can be defined as:

$$D^*_G(x|l) = \frac{p_r(x|l)}{p_r(x|l)+p_g(x|l)}. \tag{12}$$

If the optimal value of D is given, Eq. 5 can be re-expressed as:

$$\begin{aligned} C(G) &= \max_D V(D,G) \\ &= \mathbb{E}_{x\sim p_r}[\log D^*_G(x|l)] + \mathbb{E}_{z\sim p_z}[\log(1-D^*_G(W(G(z|l,c))))] \\ &= \mathbb{E}_{x\sim p_r}[\log D^*_G(x|l)] + \mathbb{E}_{x\sim p_g}[\log(1-D^*_G(x|l))] \\ &= \mathbb{E}_{x\sim p_r}[\log \frac{p_r(x|l)}{p_r(x|l)+p_g(x|l)}] + \mathbb{E}_{x\sim p_g}[\log(\frac{p_g(x|l)}{p_r(x|l)+p_g(x|l)})] \\ &= \int_x p_r(x|l)(\log 2 - \log 2)p_r(x|l) + \log\frac{p_r(x|l)}{p_r(x|l)+p_g(x|l)} \\ &\quad + (\log 2 - \log 2)p_g(x|l) + p_g(x|l)\log(\frac{p_g(x|l)}{p_r(x|l)+p_g(x|l)})\,dx \\ &= -\log 2 \int_x p_r(x|l) + p_g(x|l)\,dx + 2\cdot JSD(p_r\|p_g). \end{aligned} \tag{13}$$

Since the Jensen-Shannon divergence between the two distributions is always non-negative, and is zero only when the two distributions coincide, the global minimum of $C(G)$ is then $C^* = -\log(4)$ which can only be reached if $p_r = p_g$. In addition, it is difficult to directly compute the mutual information values, $G(z, l, c)$, between the latent code, c, and generated actions, because the posterior $P(c|G(z, l, c))$ is unknown. To solve this, an auxiliary distribution $Q(c|G(z, l, c))$ is introduced to approximate $P(c|G(z, l, c))$. Thus, the mutual

information $I(c; G(z, l, c))$ can be expressed as:

$$\begin{aligned} I(c; G(z,l,c)) &= H(c) - H(c|G(z,l,c)) \\ &= \mathbb{E}_{x\sim G(z,l,c)}\left[\mathbb{E}_{c'\sim P(c|x)}[\log P(c^{'}|x)]\right] + H(c) \\ &= \mathbb{E}_{x\sim G(z,l,c)}\left[D_{KL}(P(c^{'}|x)|Q(c^{'}|x)) + \mathbb{E}_{c'\sim P(c|x)}[\log Q(c^{'}|x)]\right] + H(c). \end{aligned} \tag{14}$$

Since the Kullback-Leibler divergence between the two distributions is always non-negative, the variational lower bound of mutual information can be defined as:

$$\begin{aligned} L_I(G,Q) &= \mathbb{E}_{x\sim G(z,l,c)}\left[\mathbb{E}_{c'\sim P(c|x)}[\log Q(c^{'}|x)]\right] + H(c) \\ &\leq I(c; G(z,l,c)) \leq H(c). \end{aligned} \tag{15}$$

Lemma 1. *For random variables X, Y and function $f(x, y)$ under suitable regularity conditions: $\mathbb{E}_{x\sim X, y\sim Y|x}[f(x, y)] = \mathbb{E}_{x\sim X, y\sim Y|x, x'\sim X|y}[f(x^{'}, y)]$[29].*

According to Lemma 1, $L_I(G, Q)$ can be easily approximated by a Monte Carlo simulation. According to Eq. 14, the lower bound becomes tight when the auxiliary distribution, Q, approaches the true posterior distribution: $D_{KL}(P(c^{'}|x)\|Q(c^{'}|x)) \to 0$. In addition, the effect of maximizing $L_I(G, Q)$ is equivalent to minimizing $D_{KL}(P_c(c)\|P_x(x))$, here $x = Q(G(z, l, c)$. Thus, $L_I(G, Q)$ can be reformulated as:

$$L_I(G,Q) = c \cdot \log Q(G(z,l,c)) + H(c). \tag{16}$$

The $D_{KL}(P(c)\|P(x))$ can be defined as:

$$\begin{aligned} D_{KL}(P_c(c)\|P_x(x)) &= -c \cdot \log \frac{Q(G(z,l,c))}{c} \\ &= -c \cdot \log Q(G(z,l,c)) + c \cdot \log c \\ &= -c \cdot \log Q(G(z,l,c)) + H(c) \\ &= -L_I(G,Q) + 2H(c). \end{aligned} \tag{17}$$

Thus, when $D_{KL}(P(c^{'}|x)\|Q(c^{'}|x)) = 0$ and $D_{KL}(P_c(c)\|P_x(x)) = 0$, the lower bound becomes tight and attains its maximum $L_I(G, Q) = H(c)$. Since L_I can be maximized with respect to Q directly and with respect to G via

the re-parametrization trick, the inclusion of L_I in the GANs objectives (as defined in Eq. 6) does not change the training procedure of the GAN model.

When the Stochastic Policy Gradient algorithm is introduced into the GAN model, the model becomes a combination of the GAN model and the reinforcement learning algorithm. The generator of the GAN model is also a parametric representation of the policy, θ. The objective of the reinforcement learning algorithm is to maximize the reward value through finding an optimal θ:

$$\max_{\theta} J(\theta) = \max_{\theta} \sum_{\tau} \log P(\tau, \theta) R(\tau), \tag{18}$$

where τ denotes the action trajectory sampled from the policy, θ; $R(\tau)$ denotes the reward value of τ. Then, the gradient information of the generator, G, is reformulated as:

$$\nabla_\theta J(\theta) = \mathbb{E}_{z\sim p_z, l\sim p_l, c\sim p_c}[\nabla(\log \prod_{i=1}^{n} G(z,l,c)_i) \cdot D(W(G(z,l,c)))]. \tag{19}$$

In fact, the approximation of $\nabla_\theta J(\theta)$ can be readily calculated using the Monte Carlo Simulation method. Thus, the parameters of the generator can be updated by:

$$\theta_{new} \leftarrow \theta_{old} + \alpha \nabla_\theta J(\theta), \tag{20}$$

where α denotes the learning rate of the generator.

Since a trajectory, τ, is sampled from the policy, θ, through Monte Carlo Simulation, τ is not fixed when the same state is encountered. Therefore, the policy, θ, has the capability to explore the working environment. This property of the stochastic policy can improve the stability of the GAN model. This is because in the original GAN model, the utilization of the Kullback-Leibler divergence for loss measurement of the model has the potential to cause the gradient to disappear and the generated samples to be non-diversified. For example, if there is no overlap between two data distributions, the Kullback-Leibler divergence of these two distributions is a constant, "$\log 2$", i.e., the gradient value being 0. In addition, due to the asymmetry of the Kullback-Leibler divergence, the Kullback-Leibler divergence is not balanced in the evaluation of the generated samples; so as to cause the generation strategy

of the generator to be conservative. However, in the proposed model, due to the exploration mechanism of the stochastic policy, even when the gradient is disappeared, the exploration mechanism can still provide potential gradient information. Therefore, the introduction of stochastic policy gradient algorithm alleviates these two limitations of the Kullback-Leibler divergence, so as to improve the performance of the proposed system in theory with empirical proof provided in the next section.

4. Experimentation

The evaluation of the proposed approach is reported in this Section. The experimental setup and training data sets are introduced first; then, the training phase and writing results are demonstrated. This is followed by the evaluation of the condition information and latent codes. Two comparison experiments are conducted to show the writing quality and stroke diversity of the proposed method.

4.1. Experimental Setup

The proposed model was applied to perform a Chinese character stroke writing task to verify its effectiveness. The robot writing platform used in the experiment is shown in Fig. 1a. The model was implemented using the Tensorflow, an open-source software library for machine intelligence. There are a number of parameters in the proposed model, and the values used in this experiment are summarized here. The value for each instance of noise ranges from -1 to 1. The number of latent codes (whose values are set to continuous) is 2. Therefore, the values for the latent codes are also in the range of [-1, 1]. The hyper-parameter λ is empirically set as 1. The learning rates for all three networks are set to 0.001.

4.2. Training Data

The training data set includes samples from two sources, and thus there are two categories of samples, including the "real samples" and the "fake samples". In particular, a fake sample was generated by the network G and the calligraphy robot system. Each such sample (x_{fake}) is comprised of the input (l) of the network G and the output image (x) of the robot system. The real samples were taken from a stroke data set, containing six different Chinese character strokes. Each sample of this data set contains a stroke image and a stroke type label. This data set contains 3,578 data instances.

Each stroke has more than 500 samples. The images of some example strokes are shown in Fig. 3. The types of strokes from a to f are: "short left-falling", "horizontal", "horizontal and left-falling", "right-falling", "long left-falling", and "vertical, turn-right and hook".

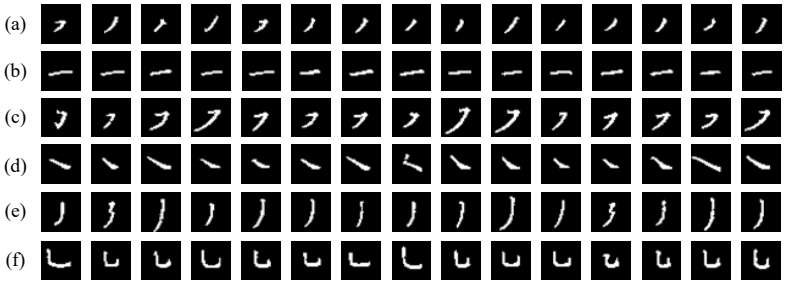


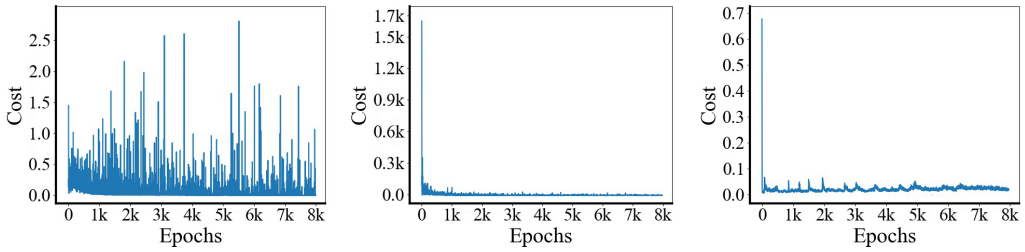
Figure 3: Illustrative training samples used in the experiment, each row shows one type of strokes with various variations.

The training samples in the network Q were generated from the G network. In each training sample, $s(c, G(z, l, c))$, c is a part of the input of network G, and $G(z, l, c)$, depending on its input c, is the output of network G. Moreover, in the training phase, the inputs c and l of network G were generated randomly.

4.3. Training Phase and Writing Results

In the model training process, the G network began to converge after 4,000 epochs. The cost change of three networks during the training process is shown in Fig. 4. The cost value of the discriminative network, calculated by Eq. 7, is shown in Fig. 4a. The discrimination ability of the D network changes in the training process is summarized as follows:

1. Before the first hundred training epochs, the cost value of the D network is about 1.4 ($2 \times \ln 0.5$). This indicates that the D network in this phase is unable to determine whether or not each sample is real or fake. Thus, the probability that the D network correctly distinguishes a sample is around 0.5.
2. After several hundreds of training epochs, the cost value of the D network quickly decreased, but the cost of the G network was kept high. In this case, the quality of the strokes generated by the G network was still not good, and the D network sped up in gaining the discriminative ability.



(a) Discriminative network. (b) Generative network. (c) Auxiliary distribution network.

Figure 4: Cost change of three networks during the training process.

3. After the cost of the G network was further reduced, the G network generated several good-quality strokes. As a result, the task complexity of the D network increased, and the cost of the D network began to increase slowly.
4. Finally, the G network reached its maximum ability to generate samples, and the training of the G network began to converge to its limit.

During this convergence process, the cost of the D network increasingly fluctuated. What is worse, for some epochs, the cost even exceeded the initial cost value (1.4), which indicates that the D network in these epochs, without discrimination ability, mistakenly identified fake samples as real or vice versa.

Fig. 4b shows that the capacity of the G network steadily increased until the maximum was reached, indicating that the policy gradient method was effective in the training process. As demonstrated in Figs. 4a and 4b, whenever the ability of the G network reached a local peak, the ability of the D network reached a local minimum. After the G network converged, the cost of the D network has reached 1.4 or more in only a few epochs, indicating that the learning ability of the D network is greater than that of the G network. Accordingly, the quality of the strokes generated by the G network may still have room for improvement.

Fig. 4c shows the Q network cost changes in the training process. The Q network convergence was very rapid. In less than one hundred epochs, the Q network has started to converge, showing that the problem the Q network faces is not complicated. Notably, after a period of training, the cost of Q suddenly increased significantly, and then quickly decreased. Accompanying this phenomenon is that the feature information presented by c is changed

after this period of Q network training. For example, first, after a training period, c learned the width feature of the short left-falling stroke. Second, the Q network trained in another period, until the cost of the Q network rose and fell quickly. After these two stages of training, the feature information represented by c was replaced by other features, such as size, length, etc. After the proposed model finished its training, the robot was able to write a variety of strokes based on the trajectory points generated by the G network.

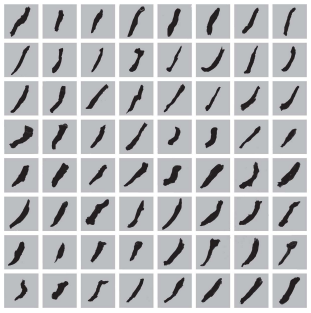
4.4. Evaluation of Condition Information

To verify the effect of condition information, l, in the model, a set of experiments were included where l was set as the label of the six strokes. In these experiments, the values of z and c were generated randomly. The experimental results are shown in Fig. 5, which confirms that the model can generate the corresponding strokes based on the label information. With the same label, the styles of the generated strokes are also different, which shows the power of the proposed approach in producing required strokes with good diversity. In addition, since the values of z and c were generated randomly, the generated stroke styles were also random.

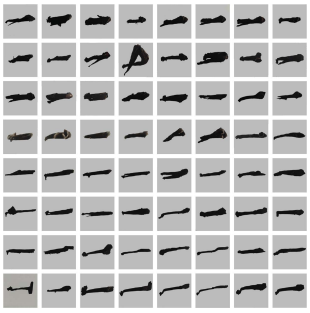
Several strokes shown in Fig. 5 are very close to those written by a human, despite a few incorrect ones. The reason for these errors can be very complex, but is related to the following: (1) Different strokes interfere with each other. Label information, l, may not be fully utilized in the G and D networks. Their classification ability for the label must be enhanced. (2) The G network is too small to fully learn to generate all six kinds of strokes. The error rate for complex strokes (i.e., "Horizontal and left-falling stroke", and "Vertical, turn-right and hook stroke") is significantly higher than that for simple strokes (including "Horizontal stroke", and "long left-falling stroke"), which indicates the insufficient generative capacity of the G network.

4.5. Evaluation of Latent Codes

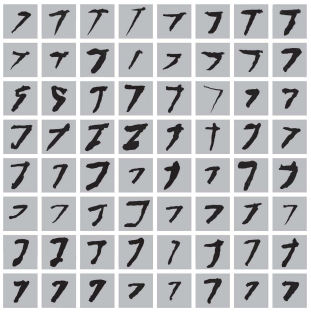
To verify the effect of the latent codes, c, the third set of experiments were designed. In these experiments, the label information, l, is fixed; the noise z, is generated randomly; and the Latent codes, c_1 and c_2, are increased from -1 to 1 with fixed steps. The experimental results are shown in Fig. 6. From this figure, it is interesting to see that the shape of a stroke went from curved to straight when the value of c_1 changed from -1 to 1; the shape of a stroke changed from thick to thin, when the value of c_2 went from -1 to 1. This illustrates that c_1 learned the curved nature of the long left-falling stroke, and



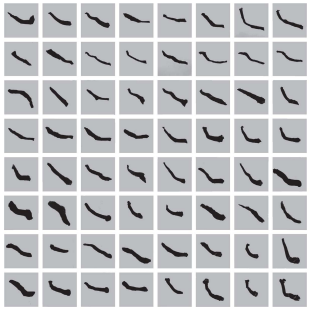
(a) Short left-falling stroke



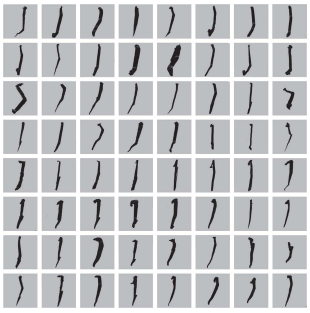
(b) Horizontal stroke



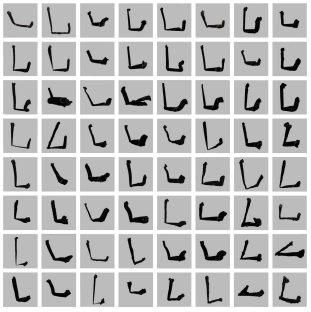
(c) Horizontal and left-falling stroke



(d) Right-falling stroke

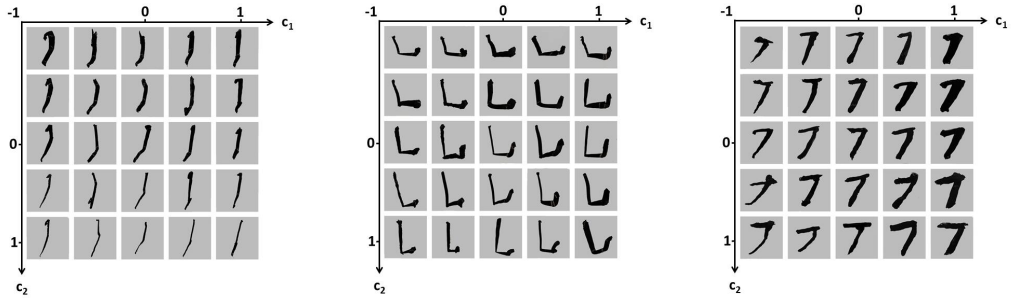


(e) Long left-falling stroke



(f) Vertical, turn-right and hook stroke

Figure 5: Writing results of all the six strokes.



(a) (b) (c)

Figure 6: The effect of stroke style changes with potential code changes. (a) Long left-falling stroke; (b) Vertical, turn-right and hook stroke; (c) Horizontal and left-falling stroke. The horizontal axis shows the change of c_1 from -1 to 1 and the vertical axis shows the change of c_2 from -1 to 1.

c_2 learned the width nature of this stroke. However, by comparing Fig. 6a, Fig. 6b and Fig. 6c, it is clear that the feature information learned by each latent code varies for different strokes. In addition, the feature information learned by a latent code c_i is also different in different training periods for the same stroke. This means that the feature information that the latent codes learned was only partially controllable. Although the learned features of the latent codes are not controllable, the variation of the stroke styles is modifiable by revising the value of the latent code.

4.6. Chinese Character Writing

The proposed stroke writing model can be combined with stroke combination methods to write Chinese characters [40], [41], [42]. The last experiments combined the proposed model with the stroke combination method as reported in the work of [43]. In this work, the stroke trajectories obtained by the human-robot interactions were retained into a stroke database, and the human gestures must match the strokes in the stroke database. Then, the robot writes the corresponding strokes on the writing board according to the human gesture. As shown in Fig. 7, based on six strokes generated by the proposed model, the Chinese character "yong" (which means forever) can be written via the stroke combination method.

Figure 7: Chinese characters are written by the proposed model in combination with other methods. The first is a printed character.

4.7. Findings

In the process of writing a stroke by a robot, according to trajectory points generated by the model, the robot sometimes exhibited several interesting behaviors. For example, the robot can use only three points to write a stroke; then, the robot uses the remaining two trajectory points to fine tune the written stroke; so that, a high writing quality stroke can be obtained. Moreover, in the proposed model, to ensure the robot writes the stroke in the center of the writing board, the trajectory points located beyond the range of the coordinate system are set as null. The robot does nothing when it is instructed by these null trajectory points. The model uses this rule unexpectedly to learn a generation skill that uses fewer trajectory points to write a stroke. Occasionally, more than two of all five trajectory points are null, and the robot uses the remaining points to write a complete stroke. In addition, the probabilities of this phenomenon and the appearance of null trajectory points are higher when writing simple strokes in writing strokes than those in writing complex strokes. This shows that the model learned to use different strategies to generate strokes of different complexity, and to choose a more effective strategy when writing simple strokes.

When the model generates a stroke with fewer than five trajectory points, the remaining points, in addition being represented as null trajectory points, are sometimes also used to describe a portion of the stroke. As shown in Fig. 5b, the stroke in the first row of the first line was generated in such a way. After the robot writes a stroke based on the three points, it refines the stroke with the remaining two points. This kind of beautification behavior also often happens when children learn to write. When children cannot correctly understand stroke order, they often write the outline of the stroke first and then further refine the stroke with more details. Similarly, in the proposed model, this phenomenon might be caused by the fact that the model does not consider the stroke order when generating stroke trajectory points.

These interesting behaviors manifested by the robot, on the other hand,

also reflects the limitations of the proposed model in terms of character stroke order. In the proposed model, the G network learns to generate trajectory points based on the reward feedback from the D network. The input of the D network is simply an image of a stroke, which does not contain any stroke order. This can be eliminated if needed, by adding the stroke order as part of the input of the D network.

The discrete latent codes also attempted to learn the label information of strokes from unlabeled training samples. In this experiment, the label information l was removed from the G and D networks, and the latent codes were changed to a 6-dimensional discrete vector. The experimental results are reported in Fig. 8. The model only successfully learned three kinds of stroke label information. Among them, three hidden codes were all learned the Short left-falling stroke. Their styles are different. The results demonstrate that the difference between the three styles of this stroke is greater than the difference between other strokes from the model point of view. In addition, the model also learned the label of a null stroke. The reason for task failure may be down to the small number of training samples; also, the difference between the same type of samples is greater than that between different types of samples.

4.8. Comparative Study

In order to prove that the effects of the proposed approach, two comparative experiments were conducted. The first experiment is to evaluate the writing quality and the second one is to evaluate the style diversity. In addition, in order to analyse the differences in mathematics, "Fréchet Inception Distance" (FID) [44] was used to measure the difference between the experimental results:

$$d^2((\boldsymbol{m}, \boldsymbol{C}), (\boldsymbol{m}_w, \boldsymbol{C}_w)) = ||\boldsymbol{m} - \boldsymbol{m}_w||_2^2 + Tr(\boldsymbol{C} + \boldsymbol{C}_w - 2(\boldsymbol{C}\boldsymbol{C}_w)^{1/2}), \quad (21)$$

where $\boldsymbol{m}$ and $\boldsymbol{C}$ denote the mean and covariance of the model samples, respectively; $\boldsymbol{m}_w$ and $\boldsymbol{C}_w$ denote the mean and covariance of the samples from real world, respectively.

4.8.1. Writing Quality Comparison

In order to verify the advantages of the proposed method compared to other methods, two similar approaches were employed here. The first comparison method is a reinforcement learning method [45], which evaluates the

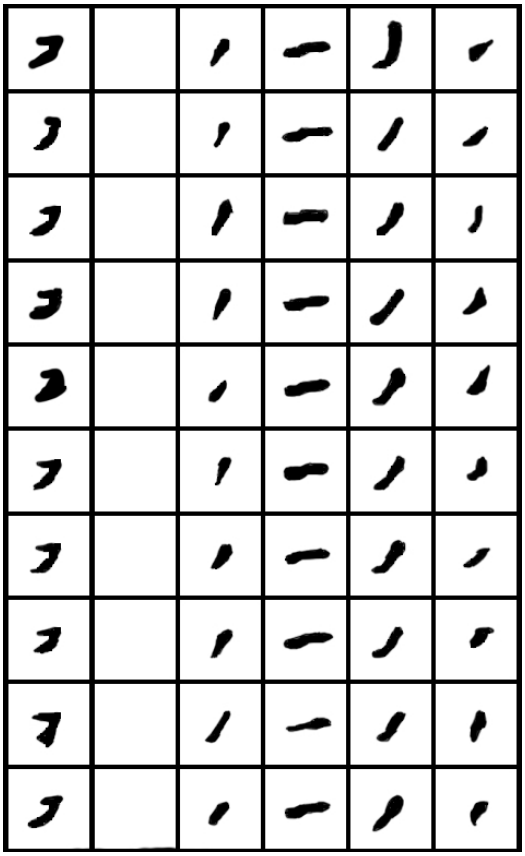


Figure 8: Experimental results of six discrete latent codes, each column represents the result of a latent code.

results of calligraphy by a manually designed reward function. Another comparison method is a traditional GAN model-based robotic writing system [35]. These two methods were trained using the same training set in this experiment to facilitate the comparison.

First, these methods are compared from the perspective of the quality of the generated strokes. The FID values between the results generated by the two comparison methods, and the employed data set for comparison are shown in Table 1. The column "Real strokes" in the table indicates the FID value between the two sub-data sets in the training set. And, the column "Proposed method" represents the FID values between the comparison data set and the test set generated by the proposed model. The last two columns of Table 1 represent the FID values of the two comparison methods, respectively. It is clear in this table, the reinforcement learning method using the artificial design reward function had the worst performance, while the methods based on the GAN model were better. This shows that the replacement of the reward function by the GAN model does improve the writing quality of the robot. In addition, compared with the traditional GAN model, the performance of the proposed model does not degrade and achieve better results on the four strokes. This shows that the introduction of condition information and latent codes in the model does not significantly reduce the performance of the model.

Stroke category	**Real strokes**	**Proposed method**	**Traditional GAN**	**Reinforcement learning**
Stroke a	11.53	53.12	56.19	88.02
Stroke b	5.41	54.80	49.62	85.89
Stroke c	16.30	49.58	64.89	92.67
Stroke d	11.77	58.04	59.28	91.69
Stroke e	15.52	57.73	57.77	94.39
Stroke f	9.26	52.70	48.49	93.70

Table 1: FID values between Various methods and the training data set.

4.8.2. Stroke Diversity Comparison

The three methods were also compared in terms of the style diversity of the generated results. To measure the diversity of the generated data, the internal FID value between the results generated by the model is calculated. This is achieved by firstly randomly dividing the generated data into

four subsets; then the FID values between every two sub-sets are calculated; from this the mean value of all FID values is computed as the internal FID value of the generated data. If the FID value between the internals of the generated data is small, the distribution of the data is generally more concentrated. And, the more concentrated the distribution of data is, the smaller the diversity of data is.

The internal FID values of the data generated by the three methods are shown in Table 2. From this table, it is clear that the FID value of the proposed method is significantly larger than those of the other two methods. This demonstrated that the hidden code designed in the model does improve the diversity of model generation results.

Stroke category	**Our method**	**Traditional GAN**	**Reinforcement learning**
Stroke a	23.43	16.20	8.19
Stroke b	14.95	10.78	5.82
Stroke c	27.61	24.35	12.88
Stroke d	24.38	15.96	9.08
Stroke e	26.67	23.41	10.73
Stroke f	21.10	14.17	7.79

Table 2: The internal FID values of the data generated by the three methods.

4.8.3. Qualitatively Comparison

Based on the above experiments, the advantages of the proposed approach have been verified. To further reveal its strengths, the proposed model is qualitatively compared with the conventional calligraphy robot methods. In addition to the two methods listed in Table 2, six extra existing methods are also employed here for qualitative comparison, with the results summarized in Table 3. From this analysis, the advantages of the proposed model can be summarised from three aspects:

1. **Simpler evaluation mechanism**: A large number of existing robotic calligraphy systems use an open-loop structure, all with the requirement of human involvement. The intensive labor requirement indeed restricts the efficiency of the model. Some other models use a closed-loop structure. Such models must contain evaluation mechanisms to

Options:	Conventional approaches:	Proposed approach:
Evaluation mechanism	Evaluate by humans [21, 25] or evaluate by an evaluation algorithm designed by algorithm engineers [12, 13, 24, 45]	Automatically build the evaluation mechanism without human intervention
Diversity	Only a few styles exist in the front database and limited to the style of the study sample [9, 12, 13, 35, 45]	A style different from the training sample can be generated. In theory, it can generate unlimited kinds.
Style control	Most methods cannot be controlled [9, 12, 13, 35, 45]	Supported by input parameters
Quality	The reinforcement learning method has larger FID distance [45]; and the GAN method demonstrates an advantage in two of the six strokes [35].	Closed to training data set.

Table 3: Summary of the qualitatively comparison with the conventional approaches.

support the training modules with clear optimization objectives or gradient information. The performance of these models is limited by the evaluation mechanisms, given the challenges in designing such mechanisms. Compared with these methods, the proposed model does not need to design an evaluation function; instead, the task of evaluating functions in the proposed work is achieved by the discriminator of the GAN network. Based on the evaluation results of the discriminator, the generator learns to generate appropriate calligraphy actions. Therefore, the discriminator can be used instead of the artificially designed evaluation function as used in conventional calligraphy robots.

2. **The diversity of stroke styles**: In traditional methods, the styles of the generated strokes are dependent on the stroke database for training. However, the styles of characters in the stroke database are limited, which leads to a limited type of stroke styles generated by the model. Although the robot model based on a closed-loop structure can remove the restriction of the stroke database to generate some novel stroke styles, the number of stroke styles is still limited. For exam-

ple, the traditional GAN model is able to generate many stroke styles, different from the database. However, only 128-dimensional Gaussian distribution-based random noise was used as the input of such models, which limits the stroke style variation, and the change is not controllable. In contrast, in the proposed model herein, the space for each trajectory point is 784, and its width is 20. In addition, the input, z, of the G network is 128-dimensional random noise; c is 2-dimensional controllable noise. Hence, the G network is able to generate a massive number of different trajectory points for each stroke. Furthermore, the proposed model can also control the type and style of a stroke through the introduction of the label information, l, and latent codes c, which is not featured by the traditional GAN method.

3. **High writing quality with simple learning system**: The goal of the robot model based on a closed-loop structure is to find the optimal solution to the evaluation function. Thus, the quality of writing fundamentally depends on the quality of the evaluation function. However, it is difficult to obtain human-level writing performance using only computational generation methods. In the GAN model, the function of the D network is to distinguish the authenticity of the input sample. The goal of G network is to generate real samples. Therefore, in theory, the G network can generate samples that human beings cannot tell the source of the sample. In addition, the G network is trained through a policy gradient method, and it is proved that the optimal solution can be infinitely approximated in theory. In conclusion, the proposed model simulates human-level writing using a simple learning structure, which is very promising in calligraphy robotics.

5. Conclusion

This paper presented a novel closed-loop calligraphy robot system based on the GAN. The model uses an end-to-end learning approach, which does not need a specifically designed evaluation function but a decision network. The training data are real stroke images from Chinese calligraphy textbooks, and the output is strokes that the robot writes on the board. The proposed model can control the type and style of a generated stroke, by adding label information and latent codes in the traditional GAN. The proposed model successfully generated strokes according to specified types and styles in the

experiments, with some of the results reaching human-level quality. In addition to controlling the type and style of generated strokes, the proposed approach can also evaluate the generated results through a neural network, which thus does not require purposely designed evaluation functions or human engineers to be involved.

The proposed model combines several GAN variants with the support of a training method adopted from reinforcement learning. Compared with the traditional GAN, the proposed approach adds condition information and latent codes to the model, which improves the controllability of the generated data. The proposed GAN model can also be used to generate other data, which requires simultaneous control of the dominant features and recessive features. Because the proposed model uses the discriminator of the GAN network instead of an evaluation function, the model has great potential in many other tasks where an explicit evaluation function is difficult to be designed such as robot dancing and painting. Moreover, this new usage of the discriminator network may also be used to support reinforcement learning. For example, it is sometimes difficult to design a suitable reward function for some tasks when using reinforcement learning; this can be potentially addressed by applying multiple GAN variants jointly as the reward function of reinforcement learning.

Of course, the proposed method also has several limitations when generating calligraphy actions. First, because the input of the discriminator is a calligraphy image, the model does not consider the order information of the generated actions and thus the order may be wrong. Second, the features indicated by the latent codes may be uncontrollable. During the training stage of the model, the features represented by the latent codes are always changing, and thus the final features may not be consistent with the expectation from human experts. Thus, there is room for improvement in the proposed approach. The larger the sample data is, the more accurate the model estimates the distribution of the sample. There are only a few thousand training samples applied in this work. In the future, if more training samples are collected, the performance of the proposed model is expected to be further improved. Also, the stroke order is not considered in the proposed work, which can be taken as system input in order to generate improved results in the future [46]. In addition, the stroke style learned by the model is semi-controllable, that is the change of style is controllable but the exact style is not; thus, it is worthwhile to investigate the precise control of the output stroke styles. For instance, the model may be able to, based on the

user request, generate a transition style between the two learned styles used in the training data sets. In addition, although only sub-ideal results were led by the use of discrete latent codes, further research effort is required to enable a robot to learn a stroke label directly from label-free samples.

Acknowledgment

The authors are very grateful to the anonymous reviewers for their constructive comments which have helped significantly in revising this work. This work was supported by the National Natural Science Foundation of China (No.61673322, 61673326, and 91746103), the Fundamental Research Funds for the Central Universities (No. 20720190142), Natural Science Foundation of Fujian Province of China (No. 2017J01128 and 2017J01129), and the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement (No. 663830).

References

[1] H. Zeng, Y. Huang, F. Chao, C. Zhou, Survey of robotic calligraphy research, CAAI Transactions on Intelligent Systems 11 (1) (2016) 15–26.

[2] F. Chao, Y. Huang, C. Lin, L. Yang, H. Hu, C. Zhou, Use of automatic Chinese character decomposition and human gestures for Chinese calligraphy robots, IEEE Transactions on Human-Machine Systems 49 (1) (2019) 47–58. doi:10.1109/THMS.2018.2882485.

[3] D. Berio, S. Calinon, F. F. Leymarie, Dynamic graffiti stylisation with stochastic optimal control, in: Proceedings of the 4th International Conference on Movement Computing, ACM, 2017, pp. 18:1–18:8. doi:10.1145/3077981.3078044.

[4] V. Mohan, P. Morasso, J. Zenzeri, G. Metta, V. S. Chakravarthy, G. Sandini, Teaching a humanoid robot to draw "shapes", Autonomous Robots 31 (1) (2011) 21–53. doi:10.1007/s10514-011-9229-0.

[5] D. Berio, S. Calinon, F. F. Leymarie, Learning dynamic graffiti strokes with a compliant robot, in: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2016, pp. 3981–3986. doi:10.1109/IROS.2016.7759586.

[6] R. J. Hendrick, C. R. Mitchell, S. D. Herrell, I. Robert J. Webster, Hand-held transendoscopic robotic manipulators: A transurethral laser prostate surgery case study, The International Journal of Robotics Research 34 (13) (2015) 1559–1572. doi:10.1177/0278364915585397.

[7] N. Huebel, E. Mueggler, M. Waibel, R. D'Andrea, Towards robotic calligraphy, in: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2012, pp. 5165–5166. doi:10.1109/IROS.2012.6386275.

[8] Y. Man, C. Bian, H. Zhao, C. Xu, S. Ren, A kind of calligraphy robot, in: IEEE International Conference on Information Sciences and Interaction Sciences, China, 2010, pp. 635–638. doi:10.1109/ICICIS.2010.5534678.

[9] S. Mueller, N. Huebel, M. Waibel, R. D'Andrea, Robotic calligraphy - Learning how to write single strokes of Chinese and Japanese characters, in: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2013, pp. 1734–1739. doi:10.1109/IROS.2013.6696583.

[10] F. Yao, G. Shao, J. Yi, Extracting the trajectory of writing brush in Chinese character calligraphy, Engineering Applications of Artificial Intelligence 17 (6) (2004) 631–644. doi:10.1016/j.engappai.2004.08.008.

[11] B. Zhao, M. Yang, H. Pan, Q. Zhu, J. Tao, Nonrigid point matching of Chinese characters for robot writing, in: IEEE International Conference on Robotics and Biomimetics (ROBIO), 2017, pp. 762–767. doi:10.1109/ROBIO.2017.8324509.

[12] D. Berio, S. Calinon, F. F. Leymarie, Generating calligraphic trajectories with model predictive control, in: Proceedings of the 43rd Graphics Interface Conference, Canadian Human-Computer Communications Society, 2017, pp. 132–139. doi:10.20380/GI2017.17.

[13] Z. Ma, J. Su, Aesthetics evaluation for robotic Chinese calligraphy, IEEE Transactions on Cognitive and Developmental Systems 9 (1) (2017) 80–90. doi:10.1109/TCDS.2016.2645598.

[14] Z. Lian, B. Zhao, J. Xiao, Automatic generation of large-scale handwriting fonts via style learning, in: SIGGRAPH ASIA 2016 Technical Briefs, ACM, 2016, pp. 12:1–12:4. doi:10.1145/3005358.3005371.

[15] F. Chao, Y. Sun, Z. Wang, G. Yao, Z. Zhu, C. Zhou, Q. Meng, M. Jiang, A reduced classifier ensemble approach to human gesture classification for robotic Chinese handwriting, in: IEEE Conference on Fuzzy Systems (FUZZ-IEEE), 2014, pp. 1720–1727. doi:10.1109/FUZZ-IEEE.2014.6891656.

[16] F. Yao, G. Shao, Modeling of ancient-style Chinese character and its application to CCC robot, in: International Conference on Networking, Sensing and Control, IEEE, 2006, pp. 72–77. doi:10.1109/ICNSC.2006.1673120.

[17] L. M. Hiatt, C. Narber, E. Bekele, S. S. Khemlani, J. G. Trafton, Human modeling for human-robot collaboration, The International Journal of Robotics Research 36 (5-7) (2017) 580–596. doi:10.1177/0278364917690592.

[18] F. Chao, Z. Wang, C. Shang, Q. Meng, M. Jiang, C. Zhou, Q. Shen, A developmental approach to robotic pointing via human-robot interaction, Information Sciences 283 (2014) 288 – 303. doi:10.1016/j.ins.2014.03.104.

[19] Y. Sun, H. Qian, Y. Xu, Robot learns Chinese calligraphy from demonstrations, in: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2014, pp. 4408–4413. doi:10.1109/IROS.2014.6943186.

[20] R. B. Warrier, S. Devasia, Iterative learning from novice human demonstrations for output tracking, IEEE Transactions on Human-Machine Systems 46 (4) (2016) 510–521. doi:10.1109/THMS.2016.2545243.

[21] M. Hussein, Y. Mohammad, S. A. Ali, Cold: A ROS package for continuous learning from demonstration teaching a robot to write, in: IEEE International Conference on Mechatronics and Automation (ICMA), 2017, pp. 651–657. doi:10.1109/ICMA.2017.8015893.

[22] J. Garrido, W. Yu, A. Soria, Human behavior learning for robot in joint space, Neurocomputing 155 (2015) 22–31. doi:10.1016/j.neucom.2014.12.068.

[23] Y. Lin, Y. Sun, Robot grasp planning based on demonstrated grasp strategies, The International Journal of Robotics Research 34 (1) (2015) 26–42. doi:10.1177/0278364914555544.

[24] M. Wang, Q. Fu, X. Wang, Z. Wu, M. Zhou, Evaluation of Chinese calligraphy by using DBSC vectorization and ICP algorithm, Mathematical Problems in Engineering 2016 (2016) 11. doi:10.1155/2016/4845092.

[25] F. Chao, F. Chen, Y. Shen, W. He, Y. Sun, Z. Wang, C. Zhou, M. Jiang, Robotic free writing of Chinese characters via human robot interactions, International Journal of Humanoid Robotics 11 (1) (2014) 1450007–1–26. doi:10.1142/S0219843614500078.

[26] F. Chersi, Learning through imitation: a biological approach to robotics, IEEE Transactions on Autonomous Mental Development 4 (3) (2012) 204–214. doi:10.1109/TAMD.2012.2200250.

[27] D. Berio, M. Akten, F. F. Leymarie, M. Grierson, R. Plamondon, Calligraphic stylisation learning with a physiologically plausible model of movement and recurrent neural networks, in: Proceedings of the 4th International Conference on Movement Computing, ACM, 2017, pp. 25:1–25:8. doi:10.1145/3077981.3078049.

[28] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, in: Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, K. Q. Weinberger (Eds.), Advances in Neural Information Processing Systems (NIPS), Curran Associates, Inc., 2014, pp. 2672–2680.

[29] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, P. Abbeel, Infogan: Interpretable representation learning by information maximizing generative adversarial nets, in: D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, R. Garnett (Eds.), Advances in Neural Information Processing Systems (NIPS), Curran Associates, Inc., 2016, pp. 2172–2180.

[30] W. Zhao, S. Wang, Z. Xie, J. Shi, C. Xu, GAN-EM: GAN based EM learning framework, 2018 arXiv:1812.00335 (Unpublished results).

[31] M. Mirza, S. Osindero, Conditional generative adversarial nets, 2014 arXiv:1411.1784 (Unpublished results).

[32] L. Yu, W. Zhang, J. Wang, Y. Yu, Seqgan: Sequence generative adversarial nets with policy gradient., in: AAAI Conference on Artificial Intelligence, 2017, pp. 2852–2858.

[33] J.-Y. Zhu, R. Zhang, D. Pathak, T. Darrell, A. A. Efros, O. Wang, E. Shechtman, Toward multimodal image-to-image translation, in: I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems (NIPS), Curran Associates, Inc., 2017, pp. 465–476.

[34] P. Lyu, X. Bai, C. Yao, Z. Zhu, T. Huang, W. Liu, Auto-encoder guided GAN for Chinese calligraphy synthesis, in: 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), Vol. 01, 2017, pp. 1095–1100. doi:10.1109/ICDAR.2017.181.

[35] F. Chao, J. Lv, D. Zhou, L. Yang, C.-M. Lin, C. Shang, C. Zhou, Generative adversarial nets in robotic Chinese calligraphy, in: Proceedings of IEEE International Conference on Robotics and Automation, IEEE, 2018, pp. 1104–1110. doi:10.1109/ICRA.2018.8460787.

[36] X.-h. Ma, Q.-z. Kong, W.-m. Ma, X.-w. Zhang, 4-DOF lettering robot's trajectory planning, Mechanical Engineering and Automation 165 (5) (2010) 161–163.

[37] F. Yao, G. Shao, J. Yi, Trajectory generation of the writing-brush for a robot arm to inherit block–style Chinese character calligraphy techniques, Advanced robotics 18 (3) (2004) 331–356. doi:10.1163/156855304322972477.

[38] A. Droniou, S. Ivaldi, O. Sigaud, Learning a repertoire of actions with deep neural networks, in: International Conference on Development and Learning and on Epigenetic Robotics, IEEE, 2014, pp. 229–234. doi:10.1109/DEVLRN.2014.6982986.

[39] I. Lenz, H. Lee, A. Saxena, Deep learning for detecting robotic grasps, The International Journal of Robotics Research 34 (4-5) (2015) 705–724. doi:10.1177/0278364914549607.

[40] H.-I. Lin, Y.-C. Huang, Visual matching of stroke order in robotic calligraphy, in: International Conference on Advanced Robotics (ICAR), 2015, pp. 459–464. doi:10.1109/ICAR.2015.7251496.

[41] H. Lifei, Z. Jing, H. Lingling, An improved algorithm for extracting the skeletons of the Chinese calligraphy, Microcomputer & Its Applications 17 (2011) 025.

[42] Z. Ma, J. Su, Stroke reasoning for robotic Chinese calligraphy based on complete feature sets, International Journal of Social Robotics 9 (4) (2017) 525–535. doi:10.1007/s12369-017-0410-2.

[43] F. Chao, Y. Huang, X. Zhang, C. Shang, L. Yang, C. Zhou, H. Hu, C.-M. Lin, A robot calligraphy system: From simple to complex writing by human gestures, Engineering Applications of Artificial Intelligence 59 (2017) 1–14. doi:10.1016/j.engappai.2016.12.006.

[44] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, S. Hochreiter, Gans trained by a two time-scale update rule converge to a local nash equilibrium, in: I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems (NIPS), Curran Associates, Inc., 2017, pp. 6626–6637.

[45] R. Wu, W. Fang, F. Chao, X. Gao, C. Zhou, L. Yang, C. Lin, C. Shang, Towards deep reinforcement learning based Chinese calligraphy robot, in: 2018 IEEE International Conference on Robotics and Biomimetics (ROBIO), 2018, pp. 507–512. doi:10.1109/ROBIO.2018.8664813.

[46] K. Sasaki, K. Noda, T. Ogata, Visual motor integration of robot's drawing behavior using recurrent neural network, Robotics and Autonomous Systems 86 (2016) 184–195. doi:10.1016/j.robot.2016.08.022.